**Writing Plugins**

Sometimes you want to make a piece of functionality available throughout your code; for example, perhaps you want a single method you can call on a jQuery selection that performs a series of operations on the selection. In this case, you may want to write a plugin.

Most plugins are simply methods created in the $.fn namespace. jQuery guarantees that a method called on a jQuery object will be able to access that jQuery object as this inside the method. In return, your plugin needs to guarantee that it returns the same object it received, unless explicitly documented otherwise.

Here is an example of a simple plugin:

**Example 8.1. Creating a plugin to add and remove a class on hover**

// defining the plugin

(function($){

$.fn.hoverClass = function(c) {

return this.hover(

function() { $(this).toggleClass(c); }

);

};

}(jQuery);

// using the plugin

$('li').hoverClass('hover');

For more on plugin development, read Mike Alsup's essential post, [A Plugin Development Pattern](http://www.learningjquery.com/2007/10/a-plugin-development-pattern). In it, he creates a plugin called $.fn.hilight, which provides support for the metadata plugin if it's present, and provides a centralized method for setting global and instance options for the plugin.

**Example 8.2. The Mike Alsup jQuery Plugin Development Pattern**

//

// create closure

//

(function($) {

  //

  // plugin definition

  //

  $.fn.hilight = function(options) {

    debug(this);

    // build main options before element iteration

    var opts = $.extend({}, $.fn.hilight.defaults, options);

    // iterate and reformat each matched element

    return this.each(function() {

      $this = $(this);

      // build element specific options

      var o = $.meta ? $.extend({}, opts, $this.data()) : opts;

      // update element styles

      $this.css({

        backgroundColor: o.background,

        color: o.foreground

      });

      var markup = $this.html();

      // call our format function

      markup = $.fn.hilight.format(markup);

      $this.html(markup);

    });

  };

  //

  // private function for debugging

  //

  function debug($obj) {

    if (window.console && window.console.log)

      window.console.log('hilight selection count: ' + $obj.size());

  };

  //

  // define and expose our format function

  //

  $.fn.hilight.format = function(txt) {

    return '<strong>' + txt + '</strong>';

  };

  //

  // plugin defaults

  //

  $.fn.hilight.defaults = {

    foreground: 'red',

    background: 'yellow'

  };

//

// end of closure

//

})(jQuery);